The Future of Engineering Education
2024 Annual Conference & Exposition
Oregon Convention Center ASEE
Portland, OR . June 23 - 26, 2024
Paper ID #41977

Apples or Oranges: A Step Back in Time to Understand Which Programming Language is for Novice Programmers

Kwansun Cho, University of Florida

Kwansun Cho is an Instructional Assistant Professor of the Department of Engineering Education, in the UF Herbert Wertheim College of Engineering. She has been teaching introductory computer programming courses for engineers. She holds two Masters' degrees in Electrical and Computer Engineering from the University of Florida and Yonsei University, specializing in speech signal processing. Her educational research interests include improved flipped classroom teaching/learning for students, and computer- or web-assisted personalized learning.

Mr. Umer Farooq, Texas A&M University

Umer Farooq is a Ph.D. student in the Multidisciplinary Engineering Department at Texas A&M University, with a focus on Engineering Education. Umer is part of the Learning Enhancement and Applications Development Lab (LEAD Lab). Umer contributes to research initiatives centered on educational, instructional, and workforce development in the manufacturing sector. His efforts align with the mission of the Texas A&M University Gulf Coast Center of Excellence (GCCoE), where he collaborates on diverse projects aimed at enhancing learning experiences for students, trainees, and professionals.

Dr. Saira Anwar, Texas A&M University

Saira Anwar is an Assistant Professor at the Department of Multidisciplinary Engineering, Texas A and M University, College Station. She received her Ph.D. in Engineering Education from the School of Engineering Education, Purdue University, USA. The Department of Energy, National Science Foundation, and industry sponsors fund her research. Her research potential and the implication of her work are recognized through national and international awards, including the 2023 NSTA/NARST Research Worth Reading award for her publication in the Journal of Research in Science Teaching, 2023 New Faculty Fellow award by IEEE ASEE Frontiers in Education Conference, 2022 Apprentice Faculty Grant award by the ERM Division, ASEE, and 2020 outstanding researcher award by the School of Engineering Education, Purdue University. Dr. Anwar has over 20 years of teaching experience at various national and international universities, including the Texas A and M University - USA, University of Florida - USA, and Forman Christian College University - Pakistan. She also received outstanding teacher awards in 2013 and 2006. Also she received the "President of Pakistan Merit and Talent Scholarship" for her undergraduate studies.

Work in Progress: Apples or Oranges - A step back in time to understand which programming language is for novice programmers

Abstract

In this work-in-progress paper, the emphasis is to understand the perceptions about which language should be the first programming language. Computer programming is a fundamental skill for novice engineers. However, over time, multiple programming languages have emerged and are being used as the first language for students. While in modern times, many schools around the globe, particularly in the USA, consider Python's syntax simplicity and versatility as a way to go, other places and traditional computer scientists consider C++'s efficiency as their choice. Similarly, many engineering schools introduce MATLAB as the first programming language. While these decisions are made at the university or departmental level, novice programmers, when they begin programming, are affected by this choice in more than one way as it helps them not only understand how to program but also carve the path for their future choices on kind of programs they will pursue (e.g., web applications, machine learning, or embedded systems). To understand which programming language may be relevant today, especially with the boom of AI technologies, we are taking a step backward to collect perceptions on which language may be suitable. For this purpose, using an open-ended questionnaire, we collected the data from 22 members of the instructional team (8 faculty members, 14 peer mentors/undergraduate teaching assistants) in a large R1 Southeastern university. More specifically, this paper answers the question: Which computer programming language should be introduced first to novice programmers? The paper's results are novel as they provide comparative insights into the viewpoints of faculty and peer mentors.

Keywords: programming language, novice programmers, language choice, faculty perspective, students' perspective

Introduction

Computer programming is a fundamental skill for Science, Technology, Engineering, and Mathematics (STEM) students for their future careers [1]. Particularly in engineering, novice undergraduate students are often introduced to computer programming courses [2] in their first or second years to develop computational thinking [3], problem-solving [4], [5] and mathematical modeling abilities [6], [7]. However, choosing which programming language should be the first introduced language in the curriculum is an open debate without consensus yet reached [8]. The consensus is particularly difficult due to many underlying factors, such as the purpose of teaching engineering students a programming language [9], future needs within the discipline [10], and current trends in the industry [11], to list a few of the factors. Due to these factors, many universities across the globe, particularly in the United States, adopt one of the two options: 1) decide on a programming language for all engineering students enrolled in the engineering degree, or 2) offer multiple options to the students and let students choose the language based on availability, and choice.

Besides these steps, the question of the first programming language constantly hammers in academic settings, and multiple perspectives have been considered [12], [13], [14]. In this paper, we re-opened this debate. We used the perspective of faculty and peer mentors (senior students who have taken programming courses and are part of instructional teams as teaching assistants) as a lens to answer the same question. More specifically, this paper provides a comparative perspective on the following research question: Which computer programming language should be introduced first to novice programmers?

Literature review

Prior literature suggests that selecting a first programming language for undergraduate students is a continuous debate with many historical viewpoints and factors. However, researchers have a consensus that learning to program is a hard skill [15], [16] and requires problem-solving abilities, which, if lacking, can present many challenges to novice programmers and engineers. Thus, the choice of language is a fundamental challenge [17], [18] as it leads to instructors' expectations from students and students' eventual performance in programming courses.

Although the literature leaves the choice of the first programming language as an open question [4], there can be differences between students' and faculty's choices. For example, a survey conducted in Australasia (Australia and New Zealand) and the UK in 2018 [19] suggests Java is dominant in the UK. However, in Australasia, students have recently shifted their choice from Java to Python. Among many reasons for the popularity of Python over other languages are that it helped them decide computer science as a major [20], simplicity, ease of learning, and readability of the language [9]. However, the faculty's choice is more multifaceted and is connected with educational objectives [10], course depth [21], the need for fundamental skills, and industry requirements [6]. Also, in some cases, the faculty may prefer their initial learned language [13].

Notably, the choice of language directly correlates with the difficulties students may face in the future, their grades, and their attitudes [14]. Although the literature supports that some of the difficulties may arise based on the course quality and effectiveness of the instruction [22], language-specific difficulties can be challenging for students, making the choice of first language more complex. Among the commonly noted language-specific challenges are syntax complexity in traditional programming languages like Java [23], code simplicity, turnaround time [24], debugging support [25], and integrating libraries and frameworks [26].

Although synthesized literature provides the foundation for understanding the preference for programming languages, it also suggests that the question is still relevant and open to debate. Using more perspective approaches may help advance the existing literature on this debate. It may provide newer insights that may be more relevant to today's time, especially with the advent of many AI technologies and machine learning approaches.

Research Design and Methods

This study follows the qualitative methodology. For theoretical validity, we used the purposive criterion sampling method [27], [28] to recruit faculty members and peer mentors (senior undergraduate students who have taken programming language courses and are involved in instructional teams as teaching assistants).

Site and Participants

We collected the data from 22 participants (8 faculty members and 14 peer mentors). The data were collected from the instructional team of a large R1 university. The background information is provided in Table 1.

Table 1. Background information of participants

	Faculty Members N= 8	Peer Mentors N = 14	Total N=22
Programming Experience			
<3 year	-	7 (50.0%)	7 (31.8%)
3-5 years	-	4 (28.6%)	4 (18.2%)
6-10 years	3 (37.5%)	3 (21.4%)	6 (27.3%)
11-15 years	1 (12.5%)	-	1 (4.5%)
>15 years	4 (50.0%)	-	4 (18.2%)
Taught Programming Languages			
Python	6 (37.5)	4 (23.5%)	9 (28.1%)
C++	4 (25.0%)	9 (52.9%)	13 (40.6%)
MATLAB	3 (18.8%)	3 (17.6%)	6 (18.8%)
Other	3 (18.8%)	1 (5.8%)	4 (12.5%)
More than one language	4 (50%)	3 (21.4%)	7 (31.8%)
First Learned Programming Language			
Python	1 (12.5%)	3 (21.4%)	4 (18.2%)
C++	2 (25.0%)	4 (28.6%)	6 (27.3%)
MATLAB	2 (25.0%)	3 (21.4%)	5 (22.7%)
Other	3 (37.5%)	4 (28.6%)	7 (31.8%)

Measures, Data Collection, and Analysis

We designed an open-ended questionnaire to collect participants' perceptions of the first programming languages. The questionnaire comprised 3 open-ended questions, questions on participants' background, and choice of first programming language. The open-ended questions were: 1) Why did you choose the selected language over the other languages? Please state your reasons. 2) What are some features of your selected language that may be helpful for beginners, and why? 3) Why other programming languages may be less helpful in comparison to your chosen language? The data was collected in Fall 2023.

The data were analyzed using descriptive statistics and qualitative content analysis using in-vivo coding to answer the research question. The comparative approach is on two bases: 1) faculty members vs. peer mentors and 2) first studied language vs. choice of the language.

Results

The results of the descriptive statistics indicate that most of the participants, 57%, suggested Python as the first language for engineers. The second choice was C++, which was selected by

26% of the participants, and the remaining 17% chose MATLAB as the initial language of choice for engineers. When compared between faculty members, Python remained the top choice among faculty members (67%) and peer mentors (50%). However, one noteworthy aspect of the results is that while 43% of the peer mentors selected C++ as the most favorable language for beginners, the faculty members considered C++ as not the initial language, and no one selected it as the language for engineering students. Another interesting choice was MATLAB, which peer mentors considered the least favorable initial language (7%), and faculty members found it to be the second-best language for engineers (33%). It is also interesting to note that out of 22 participants, 7 participants (32%; 2 faculty members and 5 peer mentors) selected the same language they studied as their first language. However, most participants favored a language not their first language. An interesting result indicates that 5 out of 5 (100%) participants who had MATLAB as their initial language suggested C++ (60%) or Python (40%) as a first language for engineers.

The results of the qualitative content analysis indicated that faculty members who chose Python as the first language for engineers acknowledged that the choice of the first language depends on the course goals. However, they also praised the language's ease of use, problem-solving, lower syntax requirements, availability of extensive resources, and relatability to natural language as key factors. For example, a faculty member stated:

"*I selected Python, but only IF the goal is for the students to understand programming in general. As an open source language, a multitude of resources are available for the students to access for help with retention and understanding. It would be simple for the students to utilize the Python language for any number of courses that they may take moving forward. I don't think the same could be said for MatLab*."

Another faculty member who had Python as their first language and proposed the same for students as well mentioned:

"It is beginner friendly language. Students will not need to take a lot of time to learn and understand the syntax but focus more on logical thinking and problem solving."

The peer mentors who chose Python mostly admired its lower syntax requirements and ease of use as primary reasons. Some peer mentors also admired Python's versatility, development environment, and availability of libraries. For example, a student mentioned:

"I feel like Python is a good beginner language, mainly because the syntax for Python is simpler than other languages such as c++. Python make it much easier to implement certain concepts, such as dynamic arrays, and Python has quite a few modules which make certain tasks much simpler to implement."

Another peer mentor with Python as their first language mentioned its versatility and industry preference. They stated:

"Python is a higher level language that is easier to learn when first introduced to programming. Additionally, if one already has some experience python is a much easier to learn language than

other alternatives. While it may not be the best foundation for learning programming as a whole, Python is immensely versatile and used widely for various purposes in the industry."

Faculty members found that when compared with C++, C++ is more complicated for engineering students, with a steeper learning curve due to its syntax structure and higher cognitive demands. For example, a faculty member stated:

"C++ seems to require more background information prior to being able to really utilize the language for real world applications. I understand that it's significantly faster than Python, but it has a much steeper learning curve, and thus may not be as suitable for new users."

Compared with MATLAB, faculty members highlighted the opposite problem than C++, which is related to being too specific to one kind of problem only. For example, a faculty member stated:

"MATLAB has the opposite problem; it tends to be too limited in its application to be useful broadly. This can be seen in the industry - which is moving away from MATLAB and ever more into Python. In my personal experience, it also provides an inferior foundation for future programming efforts."

Aligned with faculty members' thoughts, the peer mentors also felt that C++ is more complex, has many more requirements, is not designed for engineers, and requires advanced proficiency. For example, a peer mentor stated:

"They are less helpful because C++ is designed for efficient and complete programs. If you are not a computer engineer or software developer, this really wouldn't be useful. C++ is only fun if you like tedious, organized, and strict programming guidelines/techniques."

Like faculty members, peer mentors stated that MATLAB has problems opposite to C++. They suggested that MATLAB has limited capabilities, which may not be helpful for advanced skills. For example, a peer mentor mentioned:

"*MATLAB is for computations and modeling so a lot of the programming fundamentals you would like to build as an engineer are not emphasized.*"

Discussion and Conclusion

Which programming language should be introduced to novice programmers, particularly engineering students? It is an open debate and requires constant revisitation. This revisitation is essential today, where the boom of AI technologies and tools like ChatGPT have taken the world by storm. Such tools have diversified the need for engineers trained in programming, making it a necessary skill in the industry [10].

Considering the relevance and importance of the revisitation, this study presents the perceptions and reasoning of faculty members and peer mentors on the first programming language. The study results are intuitive and encouraging as most participants (peer mentors and faculty members) agreed that Python is a good choice for engineering students. These results align with existing literature that favored Python as a versatile, simple, easy-to-learn, and efficient programming language [9]. However, the results also highlighted the diminishing choice of MATLAB as the favored first programming language, even by those who had MATLAB as their first language. There could be many probable reasons for such a result, including limited uses of MATLAB, changing trends in the industry, or Python's ability to provide the same simplicity with the ability to cater to varying and divergent kinds of problems [9].

Another important result is the choice of C++, where students (peer mentors) favored C++ as the first programming language, and the faculty members found it least relevant for engineering students. The probable reasons for such results could be rooted in the complexity of C++ for novice programmers or faculty viewpoints on changing trends in the industry [6].

The study results raise an important question: Is our engineering curriculum staying relevant to fast-paced changes in industry and the world? Considering the study results, engineering schools around the globe, particularly in the USA, need to evaluate the industry needs and, if necessary, revisit their curriculum, especially for which language is most relevant for today's era and the future in sight.

As a qualitative study, its results must be viewed with certain limitations. The study only focused on faculty members and peer mentors' self-reported perceptions and did not account for their personal biases. Although we considered the first taken language in comparison to the proposed language, future studies may consider more than one mechanism for collecting these perceptions to minimize bias. Also, the sampling method of this study was based on a purposive criterion sampling approach, limited sample size, and one institution-based sample. Although it provides theoretical validity, the larger sample size and multi-institutional study may clarify the reasoning behind the choice of language.

References

[1] P. K. Chilana *et al.*, "Perceptions of non-CS majors in intro programming: The rise of the conversational programmer," *2015 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, Oct. 2015, doi: 10.1109/vlhcc.2015.7357224

[2] E. Murphy, T. Crick, and J. H. Davenport, "An analysis of introductory programming courses at UK universities," *arXiv preprint arXiv:1609.06622,* 2016.

[3] F. J. Agbo, S. S. Oyelere, J. Suhonen, and S. Adewumi, "A systematic review of computational thinking approach for programming education in higher education institutions," in *Proceedings of the 19th Koli Calling International Conference on Computing Education Research*, Nov. 2019, pp. 1-10, doi: 10.1145/3364510.3364

[4] D. Topalli and N. E. Cagiltay, "Improving programming skills in engineering education through problem-based game projects with Scratch," *Computers &amp; Education,* vol. 120, pp. 64-74, May 2018, doi: 10. 1016/j.compedu.2018.01.011

[5] S. R. Jayasekaran, U. Farooq, and S. Anwar, "Impact of extra credit for practice questions on programming students' participation and performance," in *2023 ASEE Annu. Conf. & Expo.,* 2023, doi: 10.18260/1-2--43888

[6] J. A. Lyon and A. J. Magana, "A review of mathematical modeling in engineering education," *International Journal of Engineering Education,* vol. 36, no. 1, pp. 101-116, 2020.

[7] M. A. Qureshi, M. Asif, and S. Anwar, "NewBee: Context-free grammar (CFG) of a new programming language for novice programmers," *Intelligent Automation and Soft Computing*, vol. 37, no. 1, pp. 439-453, 2023, doi: 10.32604/iasc.2023.036102

[8] S. R. Sobral, "The first programming language and freshman year in computer science: characterization and tips for better decision making," *Advances in Intelligent Systems and Computing*, pp. 162–174, 2020, doi: 10.1007/978-3-030-45697-9_16

[9] O. Iskrenovic-Momcilovic, "Learning a programming language,"*The International Journal of Electrical Engineering &amp; Education,* vol. 55, no. 4, pp. 324-333, May 2018, doi: 10.1177/0020720918773975

[10] A. J. Magana, M. L. Falk, and M. J. Reese Jr, "Introducing discipline-based computing in undergraduate engineering education," *ACM Transactions on Computing Education,* vol. 13, no. 4, pp. 1-22, Nov. 2013 doi: 10.1145/2534971

[11] M. T. dos Santos, A. S. Vianna Jr, and G. A. Le Roux, "Programming skills in the industry 4.0: are chemical engineering students able to face new problems?," *Education for Chemical Engineers,* vol. 22, pp. 69-76, Jan. 2018, doi: 10.1016/j.ece.2018.01.002

[12] E. Giangrande Jr, "CS1 programming language options," *Journal of Computing Sciences in Colleges,* vol. 22, no. 3, pp. 153-160, 2007.

[13] D. Gupta, "What is a good first programming language?," *XRDS: Crossroads, The ACM Magezine for Students,* vol. 10, no. 4, pp. 7-7, Aug. 2004, doi: 10.1145/1027313.1027320

[14] O. Ezenwoye, "What language?-The choice of an introductory programming language," in *2018 IEEE Frontiers in Education Conference (FIE)*, Oct. 2018, pp. 1-8, doi: 10.1109/fie.2018.8658592

[15] M. Menekse, X. Zheng, and S. Anwar, "Computer science students' perceived needs for support and their academic performance by gender and residency: An exploratory study," *Journal of Applied Research in Higher Education*, vol. 12, no. 5, pp. 1025-1044, 2020.

[16] S. Anwar, "Role of different instructional strategies on engineering students' academic performance and motivational constructs," Ph.D. dissertation, Purdue Univ., West Lafayette, IN, USA, 2020.

[17] A. Ahadi, R. Lister, S. Lal, J. Leinonen, and A. Hellas, "Performance and consistency in learning to program," in *Proceedings of the Nineteenth Australasian Computing Education Conference*, Jan. 2017, pp. 11-16, doi: 10.1145/3013499.3013503

[18] M. McCracken *et al.*, "A multi-national, multi-institutional study of assessment of programming skills of first-year CS students," *Working group reports from ITiCSE on Innovation and technology in computer science education*, 2001, pp. 125-180, doi: 10.1145/572134.572137

[19] Simon, R. Mason, T. Crick, J. H. Davenport, and E. Murphy, "Language choice in introductory programming courses at Australasian and UK universities," in *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, Feb. 2018, pp. 852-857, doi: 10.1145/3159450.3159547

[20] S. Davies, J. A. Polack-Wahl, and K. Anewalt, "A snapshot of current practices in teaching the introductory programming sequence," in *Proceedings of the 42nd ACM technical symposium on Computer science education*, Mar. 2011, pp. 625-630, doi: 10.1145/1953163.1953339

[21] G. Bain and I. Barnes, "Why is programming so hard to learn?," in *Proceedings of the 2014 conference on Innovation &amp; technology in computer science education*, 2014, pp. 356-356, doi: 10.1145/2591708.2602675

[22] S. Xinogalos, T. Pitner, M. Ivanović, and M. Savić, "Students' perspective on the first programming language: C-like or Pascal-like languages?," *Education and Information Technologies,* vol. 23, pp. 287-302, Apr. 2017, doi: 10.1007/s10639-017-9601-6

[23] C.-A. Lo, Y.-T. Lin, and C.-C. Wu, "Which programming language should students learn first? A comparison of Java and Python," in *2015 International Conference on Learning and Teaching in Computing and Engineering*, Apr. 2015, pp. 225-226, doi: 10.1109/latice.2015.15

[24] C. Nandi, A. Caspi, D. Grossman, and Z. Tatlock, "Programming language tools and techniques for 3D printing," in *2nd Summit on Advances in Programming Languages (SNAPL 2017)*,), vol. 71 of Leibniz International Proceedings in Informatics (LIPIcs), (Dagstuhl, Germany), pp. 10:1–10:12, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017.

[25] R. P. Medeiros, G. L. Ramalho, and T. P. Falcão, "A systematic literature review on teaching and learning introductory programming in higher education," *IEEE Transactions on Education,* vol. 62, no. 2, pp. 77-90, May 2019, doi: 10.1109/te.2018.2864133

[26] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design patterns: elements of reusable object-oriented software*. Boston: Addison-Wesley, 1994.

[27] S. B. Merriam and E. J. Tisdell, *Qualitative research: A guide to design and implementation*, 4th ed. San Francisco, Ca: Jossey-Bass, 2016.

[28] M. Q. Patton, *Qualitative research & evaluation methods: Integrating theory and practice*. Sage publications, 2014.